This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images,
please do not report the images to the
Image Problems Mailbox.

THIS PAGE BLANK (USPTO)

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 9/44 | A1

(11) International Publication Number: WO 97/42573

(43) International Publication Date: 13 November 1997 (13.11.97)

(21) International Application Number: PCT/US97/07623

(22) International Filing Date: 7 May 1997 (07.05.97)

(30) Priority Data:
08/646,415 7 May 1996 (07.05.96) US

(71) Applicant: ELECTRONIC DATA SYSTEMS CORPORATION [US/US]; 5400 Legacy Drive, M/S H3-3A-05, Plano, TX 75024 (US).

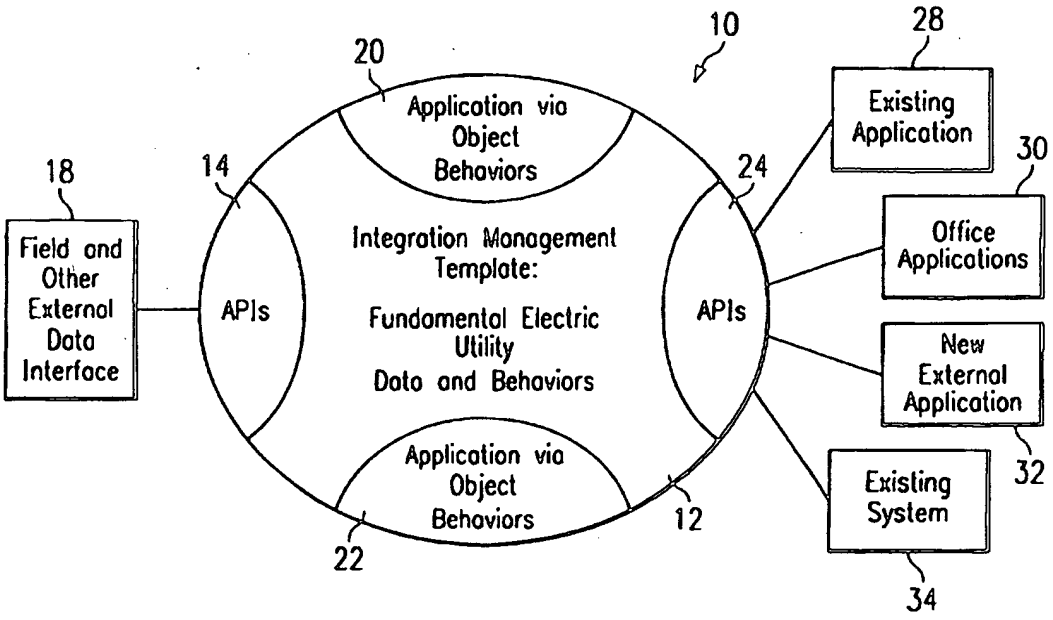
(72) Inventor: BARTON, W., Scott; 1090 York Trace, Marietta, GA 30064 (US).

(74) Agent: GRIEBENOW, L., Joy; Electronic Data Systems Corporation, 5400 Legacy Drive, M/S H3-3A-05, Plano, TX 75024 (US).

(81) Designated States: AU, CA, CN, CZ, JP, NZ, PL, European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: PERSISTENT OBJECT SUPPORT METHOD AND SYSTEM



(57) Abstract

A persistent object management system (24) supports objects (68) persistence through a common service interface associated with a computer-based integration management template (12) for modeling a complex physical system. The persistent object management system (24) includes a primary store location (64) and storing instructions (110) for storing an object (68) in the primary store location (64) for making available access to the object (68). Modifying instructions permit modifying of the object (68) while the primary store location (64) holds the object (68) as available. Retrieving instructions (130) retrieve the object (68) while the primary store location (64) holds the object (68) as available through the common service interface.

BNSDOCID: <WO 9742573A1 I >

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

BNSDOCID: <WO 9742573A1 >

PERSISTENT OBJECT SUPPORT METHOD AND SYSTEM

TECHNICAL FIELD OF THE INVENTION

The present invention relates to object-oriented computer languages and their use in modeling physical and other systems and, more particularly, to a method and system for use in a computer modeling and monitoring physical system, and even more particularly, to a method and system for supporting object persistence in an integrated infrastructure for modeling complex physical systems where the infrastructure uses object-oriented database management systems and relational database management systems through a common service interface.

BACKGROUND OF THE INVENTION

In recent years, the art of computers and software programming has evolved from the detailed line-by-line writing of source code instructions to wide-spread use of a form of programming called "object-oriented programming." Object-oriented programming is preferred over traditional methods for developing software because it facilitates designs in a tangible or physical domain. With object-oriented programming, a program is written by designing each object separately and, once the individual objects are implemented, they are connected in a coherent fashion resulting in a modular system. The software objects are also extendable and portable. This means that they can be easily reused in other designs. Most object-oriented development is performed in the C++ programming language.

Object-oriented programming objects are software entities comprising data structures and operations on the object's data. Together, these elements enable objects to model virtually any physical or real-world entity in terms of its characteristics as represented by its data elements, while its behavior may be represented by related data manipulation functions. Objects, therefore, can model concrete things such as people and computers. They can also model abstract ideas like numbers or geometrical concepts.

Planning, operations, and marketing functions of many companies or organizations have a need to integrate

planning, operations, and marketing centers as highly flexible and heterogeneous profit centers providing coordinated product and service delivery control. This need relates to information resources that assist in planning, operations and marketing.

Companies and organizations also need to address the common situation of having numerous, different systems using different business representations. These numerous, different systems often do not communicate with each other because they were built at different times using different software programming technologies and assumptions. These systems provide a variety, but not always a compatible selection, of ways to develop software, to develop additional programs and to meet additional business needs. For example, in a utility control center a great deal of data comes in from the field through other types of systems such as energy management systems. Also, within an organization there may be numerous smaller systems that support decision-making. All of these smaller systems, unfortunately, operate in isolation from other similar or possibly related smaller systems within the organization. These smaller systems employ their own application development approach for building software programs, thereby making applications development and support extremely difficult.

Another limitation of the numerous types of application programs that may exist in a company or organization are the different functions and operations

that must take place in order to make the various application programs operate and integrate with one another. The problems relate to there being different languages, different formats and different degrees of complexity within the various systems. On the other hand, eliminating all of these systems and replacing them with a new system is cost and time prohibitive.

In a single business or organization, there may be numerous different types of object-oriented database management systems and relational database management systems. If it were possible to develop an integrated management system for modeling all of the various aspects of a complex physical system, there would be a great need for an interface between the various database management or other application systems and the integrated management system. Using known programming and interfacing techniques, however, requires developing a unique interface for each of the database management or other application systems that would interface with the integrated management system. Depending on the number of database management systems and their respective language and individual complexities, this could present a formidable task.

SUMMARY OF THE INVENTION

As a result of the above, it is clear that there is a need for a method and system to support object persistence and object management functionality through a common service interface of an integration management system.

One aspect of the present invention, accordingly, includes a persistent object management system for supporting object persistence through a common service interface of a computer-based integration management template, where the integration management template models a complex physical system. The persistent object management system includes a primary store location. Storing instructions of the present invention store at least one object in the primary store location for making access to the object available for future operations. Modifying instructions modify the object while the primary store location holds the object as available. Retrieving instructions retrieve the object while the primary store location holds the object as available through the common service interface. In addition, the present invention includes releasing instructions for releasing the object through the common service interface without removing the object from the primary store location. Also, there are removing instructions that may remove the object from the primary store location, thus making it unavailable for access.

According to another aspect of the invention, there is provided a persistent object management system for supporting object persistence in a computer-based integration system for modeling a complex physical system The persistent object management system includes a common service interface for interface a plurality of application program with the computer-based integration system. The primary store location associates with the common service interface. Storing instructions associate with the common service interface for storing at least one object in the primary store location for making available access to the at least one object. Modifying instructions also associate with the common service interface for modifying the at least one object while the primary store location holds the at least one object as available. In addition, retrieving instructions associated with the common service interface for retrieving the at least one object while the primary store location holds the at least one object as available through the common service interface.

A technical advantage of the present invention is that it supports object persistence and object management functionality through a common service interface that allows for pluggable persistence of objects in an integration management system. The present invention supports numerous database management products such as object-oriented database management systems and relational database management systems.

Another technical advantage of the present invention is that it provides various storage locations. For example, the present invention includes a primary store location as the main persistence store area that accommodates all persistent object management operations. This may be, for example, an external database management system. An archival store location provides an off-line read-only store location that may be under the control of a database administrator. In at least one embodiment, an extended store location provides an on-line ready store that may be under user control for permitting extended, but not archival storage, without the immediate access requirements that a primary store location must satisfy.

Another technical advantage of the present invention is that it provides the ability to not display or communicate any changes that have not been successfully committed to the persistent store collection of at least the primary store location. This assures that the integration management template properly communicates the status of objects requiring persistence support.

Another technical advantage of the present invention is that it provides for statements from a data dictionary library associated with a related integration management template to provide the necessary commands for interfacing with a relational database management system. The present invention further generates language statements from the integration management template data

dictionary to support objects being accessed by the persistent store location.

BRIEF DESCRIPTION OF THE DRAWINGS

For more complete understanding of the present invention and the advantages thereof, reference is now made to the following description, which is to be taken in conjunction with the accompanying drawings in which like reference numerals indicate like features and wherein:

FIGURE 1 provides a conceptual view of the integration operations of an integration management template that the present invention supports;

FIGURE 2 provides an exemplary object model diagram for the initialize collection operation of the present invention;

FIGURE 3 illustrates an exemplary object model diagram for the initialize base operation of the present invention;

FIGURE 4 illustrates an exemplary object model diagram for the create object operation of the present invention;

FIGURE 5 shows an exemplary object model diagram for the store object operation of the present invention;

FIGURE 6 depicts an exemplary object model diagram for the retrieve object operation of the present invention;

FIGURE 7 illustrates an exemplary object model diagram for the retrieve all operation of the present invention;

FIGURE 8 portrays an exemplary object model diagram for the remove object operation of the present invention;

FIGURE 9 is an exemplary object model diagram for the remove all operation of the present invention;

FIGURE 10 provides an exemplary object model diagram for the delete object operation of the present invention;

FIGURE 11 illustrates an exemplary object model diagram for the release object operation of the present invention;

FIGURE 12 depicts an exemplary object model diagram for the release all operation of the present invention; and

FIGURE 13 provides an exemplary object model diagram for the update object operation of the present invention.

DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

The preferred embodiments of the present invention are illustrated in the FIGUREs wherein like numerals are used to refer to like and corresponding parts of the various drawings.

U.S. Patent Application Serial No. (EDS 34-95-011), entitled "Integration Management Template Method and System," having common inventorship herewith and commonly assigned to Electronic Data Systems describes and claims an example of an integration management template capable of effectively employing the present invention. The following description of FIGURE 1 provides a high-level description of the invention of that U.S. Patent application to establish one appropriate context for practicing the present invention. Note that although the embodiment of FIGURE 1 relates to an electric utility, the present invention should in no way be limited to such an illustrative example. The present invention may be applied to numerous types of physical systems that include complex data and behaviors among complex parameters.

Within integration management template 12, application program interfaces (APIs) 14 provide interfaces to field and other external data. This may include real-time data, inter-site data. Electronic data interchange (EDI) block 18 indicates the various field or other external data interface functions that may relate to APIs 14. Also, integration management template 12 may

include application programs via object behaviors 20 and 22 that relate specifically to the data and behaviors within integration management template 12.

Application program interfaces (APIs) 24 provide the necessary interface with existing application programs 26, office applications 30, new external applications 32 and existing systems 34, as well as other types of programs that may be part of the physical system (not explicitly shown in FIGURE 1). The present invention provides a persistence support API within integration management template 12 that permits storing one or more objects in an existing database without the need to know the existing database to a significant degree of detail.

Integration management template 12 provides a data management schema using an object model system that includes graphical user interfaces, application tool sets, and APIs such as APIs 14 and 24 for model, historical, and real-time information that a physical system may use in operations of planning and operating computer systems. Integration management template 12 uses or interoperates with a variety of object and relational database management systems to simultaneously provide an open architecture and eliminate a customer's reliance on a single database vendor.

Integration management template 12 also provides the key business processes that system operations and planning departments undertake by maintaining a common user interface and a central data repository for use by

several applications. With the common service interface, users my easily access different applications that already exist within the company or organization. This makes it possible for users to run separate applications using the same data and to transfer results from one application to another, all without leaving integration management template 12.

Integration management template 12 contains a consistent set of tools for use by all applications that are integrated into or with integration management template 12, preferably including tools for editing and viewing data or results, tools for studying case maintenance, tools for system administration, and tools for database management. With these various sets of tools, users may quickly and easily jump from one application to another to achieve desired planning, operations, and marketing results. Because the user interface is consistent for all applications, users skilled with these tools for using one application may readily apply the tools to another application through the common service interface of the present invention.

All applications within integration management template 12 use the same central database or data. All data and results from one application are immediately available for use by another application. Analysis requiring the user to run a series of applications and transfer the results from one to another is accomplished using integration management template 12.

Integration management template 12 includes a domain object model and a set of services that integrate various applications for using field and other external data. One such service is that integration management template 12 provides a centralized environment for application development that eliminates the problem of discarding work done in one application when making a change on another application. By establishing a centralized environment for application program development, integration management template 12 permits communication among multiple databases through a common service interface to develop a single representation of the physical system in a unified way. Moreover, as yet another service, integration management template 12 provides a single, but flexible, way of asking for information or data regarding the model physical system, irrespective of the particular database of origin for the data.

One of the important aspects of integration management template 12 relates to the proper storage and accessing of objects to make them useable by the many different applications. In other words, it must be determined what objects should be "persistent" and how should they be made persistent. FIGUREs 2-13, therefore, illustrate object model diagrams of the persistent object management features of the present invention. In understanding the object model diagrams of FIGUREs 2-13, it is helpful to understand the life cycle of persistent

object management that the present invention provides. The method and system of the present invention begins by initializing the collection of objects and initializing the base operations. FIGUREs 2 and 3, respectively, illustrate these operations. FIGURE 4 depicts the create object operation of the present invention. FIGURE 5 shows the next operation by using object model diagram, of storing an object.

Once the initialization, creation and storing operations take place in sequence, other operations may take place either separately or in combination. For example, the present invention permits retrieving an object or retrieving all objects, as FIGUREs 6 and 7 describe. FIGUREs 8, 9, and 10, respectively, show the removal operations of removing an object, removing all objects, and deleting an object. Release operations that the present invention provides appear at the release object model diagram of FIGURE 11 and the release all model diagram of FIGURE 12. FIGURE 13 illustrates the update object that the present invention provides, also using an object model diagram. In addition, the present invention provides for archiving and querying a primary storage location to determine the existence of an object.

Return now to FIGURE 2 which illustrates the persistent collection operation of the present invention. At point 52 where an initialize collection command uses the parameters of class type, owner identification, and table name to access persistent collection object 54.

Initialize collection operation 50 initializes a persistent collection to reference an external database. The persistent object management interface of the present invention reads the supplied parameters of class type, owner identifier, and the optional table name parameter to initialize a persistent collection for storing persistent objects identified by the class type and the owner identifier. In the initialize collection operation, a relational database management system table may be specified. The persistent object management operation of the present invention assigns a unique identifier to persistent collection 54 and creates a new process status parameter value that the persistent object management system assigned to persistent collection 54.

FIGURE 3 illustrates the initialize base operation of the present invention for initializing the state of a set of server system objects based upon a selection criterion from parameter value relating to an external database. The database operation reads selected criteria at point 62 to execute an operation leading to external database 64. In the preferred embodiment, external database 64 serves as a primary store location for the persistent objects. External database 64 initiates a status query command to generate queried results, as path 66 depicts. For this step, external database 64 executes a secured or privileged method to initialize base objects upon system start-up. The status query statement retrieves the state of instance variables that satisfy

the selection criteria of each in persistent collection 54. For each instance of a persistent object returned from the query at path 66, the present invention creates a new persistent object 68, as path 70 indicates. The retrieve state variables generate a results statement to the new persistent object, as path 72 indicates, and this adds the persistent object to its associated persistent collection 54, as path 74 indicates. Path 76 goes to process status object 78 to reflect the creation of a new process status. Path 80 goes to new process status object 78 to record the step of setting the status of process status object to record and report the outcome of the initialize base operation.

FIGURE 4 depicts the create object operation of the present invention. Beginning at point 92, the create object operation accesses persistent collection object 94 to create a new persistent object 68. Path 94 depicts the step of creating an object of the type selected by persistent collection 54. Path 96 indicates the step of assigning a unique object identifier 98 to the new persistent object 68, but not adding new persistent object 68 to persistent collection 54. Path 100 depicts the step of setting the status of the operation in process status object 78. The create object operation of FIGURE 4 then returns persistent object 68 to the requesting agent. In other words, therefore, the create object operation creates transient persistent object 68 with an initial state. This results in a persistent

object being created, initialized and returned to the requesting agent.

FIGURE 5 shows the store object operation of the present invention. Beginning at point 112 a store object operation receives a store object command is received that identifies a particular persistent object to store in persistent collection 54. Persistent collection object 54 then receives a status query language (sql) statement to insert the state of instance variables of the persistence object into the relational database system, as path 114 to persistent object 68 indicates. Path 116 indicates the step of submitting the sql insert statement to be executed by the relational database management system as external database 64. If the execution is valid, then path 118 relates to the step of committing the inserting transaction and adding the persistent object 68 to persistent collection 54. If the execution is not valid, then the insertion transaction rolls back to point 112. Path 120 leads to the step of setting status object 78 to indicate the status of the operation.

The store object operation of FIGURE 5, therefore, stores a transient object's state in external database 64 and the persistent object management system receives a persistent object. The persistent object management system changes external database 64, persistent collection 54, and process status object 78. The store object operation assumes that the persistent object

already exists in the persistent collection 54 or in external database 64. The result is thus if the persistent object is successfully stored in external database 64, then the persistent object 68 is added to persistent collection 54 and process status object 78 is set to valid. Otherwise, if persistent object 68 exists in external database 64, then process status 78 is set to valid. If neither of these can occur, process status object 78 is set to failure.

FIGURE 6 shows the retrieve object operation according to the present invention. The retrieve object operation begins by persistent collection 54 receiving from point 132 a retrieve object command that includes certain selection criteria. Persistent collection object 54, as path 134 indicates, generates a sql statement to select the state of instance variables based upon selection criteria from external database 64. Path 136 indicates the submission to external database 64 of a sql select statement to be executed by external database 64. In path 138 and for each object state retrieved, there is created a new persistent object 68, upon determining that the execution is valid. The new persistent object arises from the state information obtained from external database 64. Path 140, however, indicates that there is no creation of a unique object identifier. Path 142 indicates the step that the retrieve object operation performs of adding the new persistent object 68 to persistent collection 54. If the execution is invalid,

however, the process is terminated. Path 144 indicates the step of setting process status object 78 to indicate the outcome of the operation.

The retrieve object operation of the present invention, therefore, retrieves a set of object states based upon a selection criteria from external database 68 and makes them available for access. Upon being supplied with selection criteria at path 138, FIGURE 6 shows the operation of changing persistent collection object 54 and process status object 78, as well as creating a new persistent object 68. In the retrieve object operation, persistent object states that currently exist in persistent collection 54 are replaced. The states of the persistent objects that match the selection criteria from point 138 are retrieved from external database 64 and added to persistent collection 54. This makes the persistent objects available for access. Moreover, process status object 78 is set to valid or set to failure, depending on the success of the retrieve object operation.

FIGURE 7 shows the retrieve all operation of the present invention. FIGURE 7 shows that upon persistent collection object 54 retrieving from point 152 a retrieve all command, persistent collection object 54, along path 154, submits a sql statement to be executed by external database 64 to select all persistent object states. Along path 156, if the execution is valid, the retrieve all operation of FIGURE 5 creates new persistent object

68 from the state information obtained from external database 64. Path 158, however, indicates that there is preferably not a unique object identifier assigned to new persistent object 68. Path 160 corresponds to adding the new persistent object 68 to persistent collection 54. On the other hand, if execution is invalid, the retrieve all operation is terminated. Path 162 shows the step of setting in process status object 78.

In other words, the retrieve all operation of FIGURE 7 releases all object states for this collection from external database 64 and makes them available for access. The retrieve all operation changes persistent collection 54, process status 78, and generates new persistent object 68, and assumes that retrieve persistent objects states that currently exist in persistent collection are replaced. The result is that the states (0 through n) of all persistent objects are retrieved from external database 64 and added to persistent collection 54. This makes the persistent object 68 available for access, as well as setting process status object 78 to valid. If this cannot occur, then process status object 78 is set to failure.

FIGURE 8 shows the remove object operation of the present invention. Upon persistent collection 54 receiving via 172 a remove object command that includes selection criteria, the remove object operation uses path 174 to get the sql statement for deleting the state of instance variables based upon selection criteria from

BNSDOCID: <WO__9742573A1_I_>

external database 64. Path 176 indicates submission of the sql to a delete statement to be executed by external database 64. If execution is valid, then step 178 indicates the step of committing the delete transaction by finding each object in persistent collection 54 that satisfies the selection criteria and then removing such objects from persistent collection 54. If no objects satisfy the selection criteria, process flow rolls back the delete transaction. Path 180 indicates the step of setting the process status object 78 to indicate the outcome of the remove object operation.

The remove operation of FIGURE 8, therefore, as FIGURE 8 illustrates, removes a set of object states based on a selection criteria from persistent collection object 54 and external database 64. When the remove object operation receives supplied selection criteria, it changes external database 64, persistent collection 54 and process status object 78. Those persistent object states matching selection criteria from the remove object command are removed from external database 64 and from persistent collection 54. In response, the status object 78 is set to valid. If this cannot occur, process status object 78 is set to failure.

FIGURE 9 shows the remove all operation of the present invention. Upon persistent collection 54 receiving a remove all command, p 194 indicates the remove all operation submitting t sql statement to be executed by external database 64 t remove all persistent

object states. If path 196 indicates execution, then the remove all operation commits the delete transaction and removes each object in persistent collection 64. Otherwise, the transaction is rolled back and terminated. Path 198 indicates the step of setting process status object 78 to reflect performance of the remove all operation.

In other words, the remove all operation removes all object states from persistent collection 54 and external database 64. The remove all operation changes external database 64, persistent collection 54, and process status object 78. The result is that all persistent object states are removed from external database 64 and from persistent collection 54. Moreover, process status of 78 is set to valid. If this cannot occur, process status object 78 is set to failure.

FIGURE 10 illustrates the delete object operation of the present invention. Persistent collection 54 receives the delete object command via point 202 which specifies the persistent object for deletion. Path 204 indicates the step of getting the sql statement to delete the particular state of the persistent object from external database 64. Path 206 indicates submitting the sql delete statement to be executed by external database 64. In path 208, and if the execution is valid, the delete object operation commits the delete transaction and removes the persistent object from persistence collection 54. If this cannot occur, roll back of the delete object

operation occurs. Path 210 indicates the step of setting process status object 78 according to the outcome of the operation.

In other words, the delete operation option of FIGURE 10 deletes an object state from persistent collection 54 and external database 64. Upon being supplied the persistent object to delete, the delete object operation changes external database 64, persistent collection 54, and process status 78. The result is that the identified persistent object is deleted from external database 64 and from persistent collection 54. Moreover, process status object 78 is set to valid. If the delete object operation cannot occur, process status object 78 is set to failure.

FIGURE 11 describes the release object operation using model object diagram 220. Persistent collection 54 receives from point 222 a release object command that includes specific selection criteria. Path 224 then indicates the step of finding each object in persistent collection 54 that satisfies the selection criteria and removing it from persistent collection 54, but not external database 64. Path 226 indicates the step of setting process status object 78 to describe the outcome of the release object operation. In other words, the release object operation releases a set of object states from persistent collection 54, based on a selection criteria to make them unavailable for access. The

release object operation changes persistent collection object 54 according to the matching persistent objects.

FIGURE 12 shows operation of the release all operation. Persistent collection 54 receives from point 232 a release all command. Path 234 indicates the step of removing each persistent object from persistent collection 54, but not from external database 64. Path 236 indicates the step of setting process status object 78 according to the outcome of the release-all operation. In other words, the release all operation of the present invention releases all object states from persistent collection 54, making them unavailable for access.

FIGURE 13 shows the update object operation of the present invention. Persistent collection 54 receives from point 242 an update object command that supplies an identified persistent object. The update object operation then, along path 244, gets the sql statement for a transaction to delete the existing object state of the persistent object and inserts the new state of the persistent object. The persistent object preferably retains its unique object identifier. Path 246 indicates the step of submitting the sql statement to be executed by external database 64. If execution is valid, then the update object operation commits the delete/insert transaction. If the execution is not valid, then the update object operation rolls back the delete/insert transaction. Path 248 correspond to the step of setting

process status object 78 to reflect the outcome of the update object operation.

In other words, the update object operation updates an existing object state in external database 64 in response to a supplied persistent object identifier. The update object operation changes external database 64, persistent collection 54, and process status object 74. The update object operation assumes that the persistent object exists in the persistent collection 54 and external database 64. The result is that if the persistent object is the correct type for this collection, then the modified state of the persistent object is updated in external database 64, and process status 78 is set to valid. Otherwise, if the persistent object variable is not the proper type, then process status object 78 is set to invalid. If none of these occur, process status object 78 is set to indicate a failure of the update object operation.

In addition to the above described operations, the present invention also provides the ability to transfer between a persistent collection location and an archival location. The archival operation permits archiving a set of objects from an external database management system to an archival database management system. The archival operation of the present invention reads the supplied version, and changes external database 64, an archival database, and process status object 78. In the archiving operation, the object states of persistent objects with

version dates prior to the current version date are copied from external database 64 to an archival database. The archived object states are deleted from external database 64 and, if successful, the transaction is committed and process status object 78 is set to valid. If this does not occur, the transaction is rolled back, and process status object 78 is set to failure.

To retrieve an object from an archival database, the present invention provides an archival retrieving operation to copy a set of archived objects from the archived database to external database 64. The archive retrieval operation of the present invention reads supplied version date values and the archival database identifier. In response to these inputs and in performing the archived retrieval operation, the present invention changes external database 64, the archival database, and process status object 78. The result is that object states from persistent objects with version dates prior to the current version date are copied from the archival database to the external database 64. If successful, the transaction is committed and process status object 78 is set to valid. If this cannot occur, then the archive retrieval operation is rolled back, and process status object 78 is set to failure.

The present invention, therefore, supports object persistence and object management operationality through a common service interface that allows for a pluggable persistence engine. The present invention supports

BNSDOCID: <WO___9742573A1_I_>

numerous database management products such as object-oriented database management systems and relational database management systems by providing a primary store location as the main persistent store area, but accommodates all persistent object management operations. An archival store location provides an off-line read-only store that may be under the control of a database administrator. In at least one embodiment, an extended store location provides an on-line ready store that may be under user control.

Upon starting up of the integration management template, the present invention initializes a base domain objects for the integration management template from their associated persistent store locations and makes them available for access. The present invention provides the ability to not display or communicate any changes that have not been successfully committed to the persistent store of at least the primary store location. The present invention also provides for statements from a data dictionary library associated with a related integration management template to provide the necessary commands for interfacing a relational database management system.

Thus, in one embodiment, a system for modeling a complex physical system comprises a source of external data relating to the complex physical system, a plurality of database management system for use in modeling selected aspects of the complex physical system, and an integration management template for integrating the plurality of database management systems. The integration management template interfaces the source of external data and further comprises a common service interface for supporting object persistence and object management. The common service interface comprises of primary store location associated with the common service interface, storing

instructions associated with the common service interface for storing at least one object in the primary store location for making available access to the at least one object, modifying instructions associated with the common service interface for modifying the at least one object while the primary store location holds the at least one object as available, and retrieving instructions associated with the common service interface for retrieving the at least one object while the primary store location holds the at least one object as available through the common service interface. Such system for modeling a complex physical system may further include an archival store location for providing an off-line read-only object store location and migrating instructions associated with the common service interface for migrating the at least one object between the primary store location and the archival store location. Such system may also include a database management system associating instructions associated with the common service interface for modularly associating the common service interface with a plurality of database management systems. The integration management template may include a data dictionary, the data dictionary comprising a plurality of statements, the persistent object management system further including communicating instructions associated with the common service interface for communicating the statements from the data dictionary. Moreover, the integration management template may comprise a base domain for controlling the at least one object, the persistent object management system comprising initializing instructions associated with the common service interface for initializing the at least one object from the primary store location upon starting the integration management template.

Although the invention has been described in detail herein with reference to the embodiments, it is to be understood that this description is by way of example only and is not to be construed in a limiting sense. It is to be further understood, therefore, that numerous changes in the details of the embodiments of the invention and additional embodiments to the invention will be apparent to, and may be made by, persons of ordinary skill in the art having reference to this description. It is contemplated that all such changes and additional embodiments are within the spirit and true scope of the invention as claimed.

WHAT IS CLAIMED IS:

1. A method for supporting object persistence in association with a computer-based integration system for modeling a complex physical system, said object persistence supporting method comprising the steps of:

interfacing a plurality of application programs with the computer-based integration system;

storing through the common service interface at least one object in a primary store location, said primary store location providing immediately available access to said at least one object, said common service interface;

modifying through said common service interface said object while said at least one object remains in said primary store location; and

retrieving through said common service interface said at least one object while said primary store location holds said at least one object as available through said common service interface.

2. The method of Claim 1, further comprising the step of releasing said at least one object through said common service interface without removing said at least one object from said primary store location.

3. The method of Claim 1, further comprising the step of removing said at least one object from said primary store location for making said at least one object unavailable for access.

4. The method of Claim 1, further comprising the step of migrating said at least one object between said primary store location and an archival store location, said archival store locating for providing an off-line read-only object store location.

5. The method of Claim 1, further comprising the step of initializing said at least one object from said primary storage location upon starting said integration management template, said integration management template comprising a base domain for controlling said at least one object.

6. The method of Claim 1, further comprising the step of communicating a change to said at least one object only upon storing said at least one object in said primary store location.

7. The method of Claim 1, further comprising the step of communicating statements relating to said object persistence supporting method from statements included with a data dictionary, said data dictionary associated with integration management template.

8. The method of Claim 1, further comprising the step of associating said common service interface with a plurality of database management systems for modularly interfacing said computer-based integration management template with plurality of database management systems.

9. A persistent object management system for supporting object persistence in a computer-based integration system for modeling a complex physical system, said persistent object management system comprising:

a common service interface to interface a plurality of application programs with said computer-based integration system;

a primary store location associated with said common service interface;

storing instructions associate with said common service interface for storing at least one object in said primary store location for making available access to said at least one object;

modifying instructions associated with said common service interface for modifying said at lease one object while said primary store location holds said at least one object as available; and

retrieving instructions associated with said common service interface for retrieving said at least one object while said primary store location holds said at least one object as available through said common service interface.

10. The persistent object management system of Claim 9, further comprising releasing instructions associated with said common service interface for releasing said at least one object through said common service interface without removing said at least one object from said primary store location.

11. The persistent object management system of Claim 9, further comprising removing instructions associated with said common service interface for removing said at least one object from said primary store location for making said at least one object unavailable for access.

12. The persistent object management system of Claim 9, further comprising:

an archival store location for providing an off-line read-only object store location, and

migrating instructions associated with said common service interface for migrating said at least one object between said primary store location and said archival store location.

13. The persistent object management system of Claim 9, wherein said integration management template comprises a base domain for controlling said at least one object, said persistent object management system further comprising initializing instructions associated with said common service interface for initializing said at least one object from said primary store location upon starting said integration management template.

14. The persistent object management system of Claim 9, further comprising communicating instructions associated with said common service interface for communicating a change to said at least one object only upon said storing instructions storing said at least one object in said primary store location.

15. The persistent object management system of Claim 9, wherein said integration management template comprises a data dictionary, said data dictionary comprising a plurality of statements, said persistent object management system further comprising communicating instructions associated with said common service interface for communicating said statements from said data dictionary.

16. The persistent object management system of Claim 9, further comprising database management system associating instructions associated with said common service interface for modularly associating said common service interface with a plurality of database management systems.

17. A system for modeling a complex physical system, comprising:

a source of external data relating to said complex physical system;

a plurality of database management systems for use in modeling selected aspects of said complex physical system; and

an integration management template for integrating said plurality of database management systems, said integration management template interfacing said source of external data and further comprising;

a common service interface for supporting object persistence and object management, said common service interface comprising:

a primary store location associated with said common service interface;

storing instructions associated with said common service interface for storing at least one object in said primary store locating for making available access to said at least one object;

modifying instructions associated with said common service interface for modifying said at least one object while said primary store location holds said at least one object as unavailable; and

retrieving instructions associated with said common service interface for retrieving said at least one object while said primary store location holds at least one object as available through said common service interface.

18. The system of Claim 17, further comprising releasing instructions associated with said common service interface for releasing said at least one object through said common service interface without removing said at least one object from said primary store location.

19. The system of Claim 17, further comprising removing instructions associated with said common service interface for removing said at least one object from said primary store location for making said at least one object unavailable for access.

20. The system of Claim 17, further comprising communicating instructions associated with said common service interface for communicating a change to said at least one object only upon said storing instructions storing said at least one object in said primary store location.

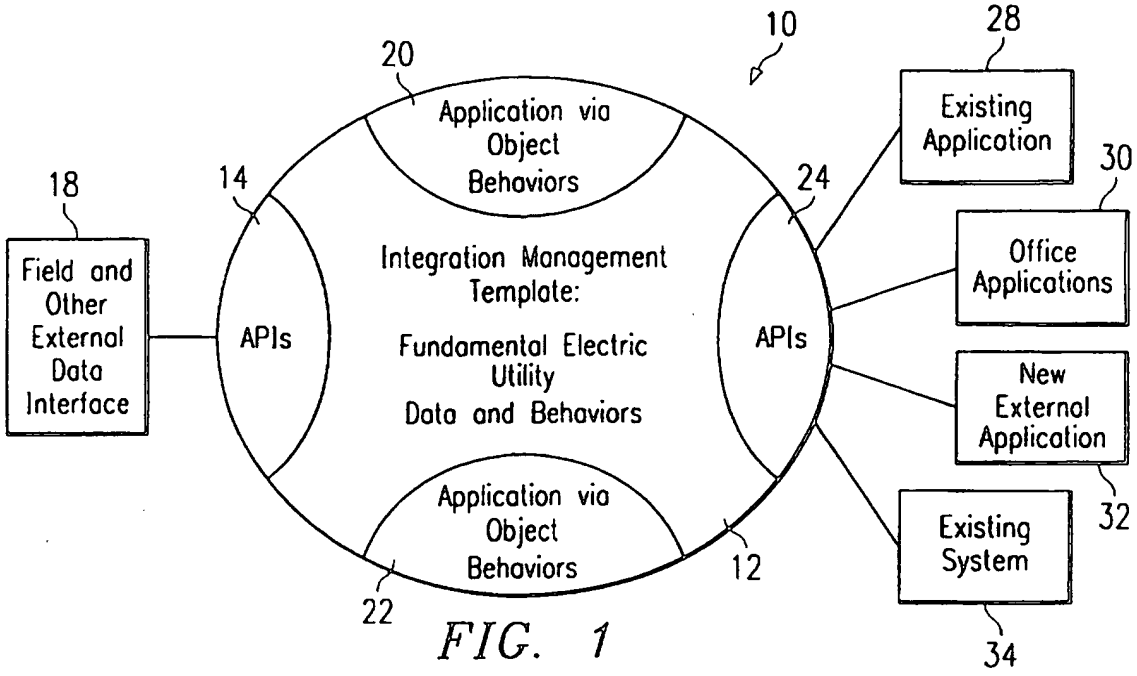


FIG. 1

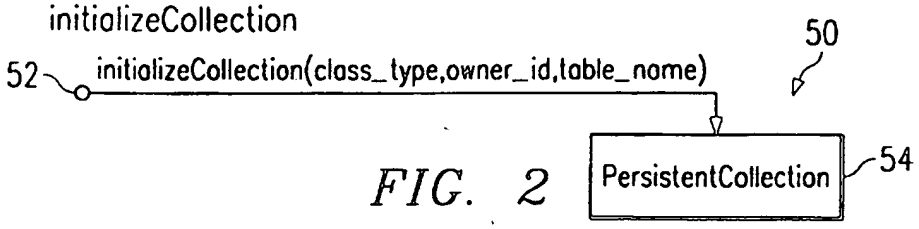


FIG. 2

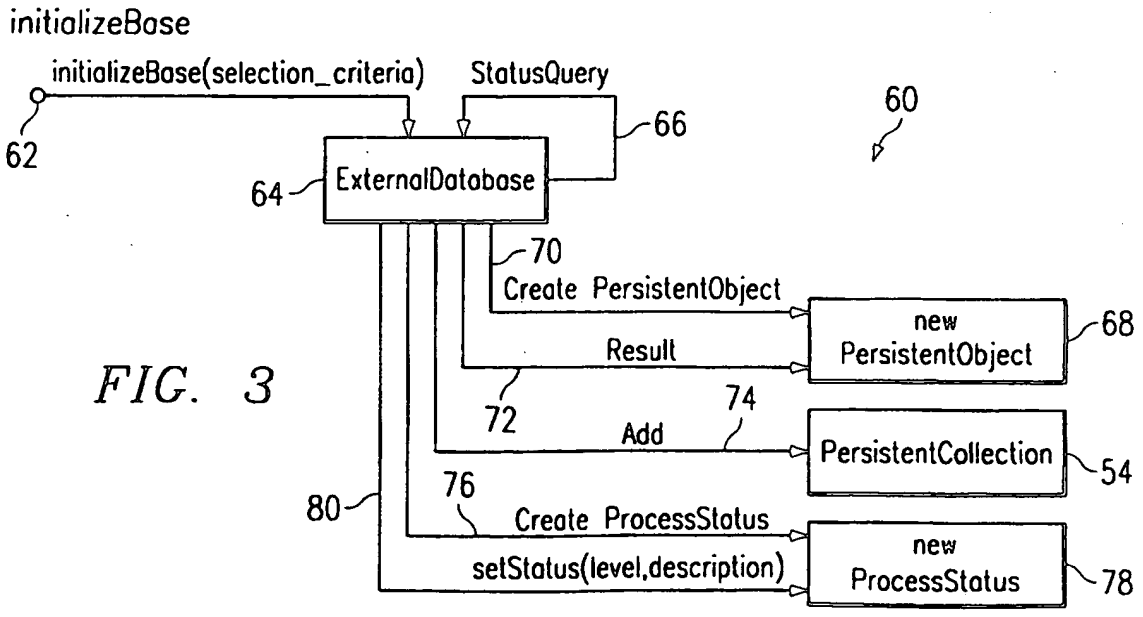


FIG. 3

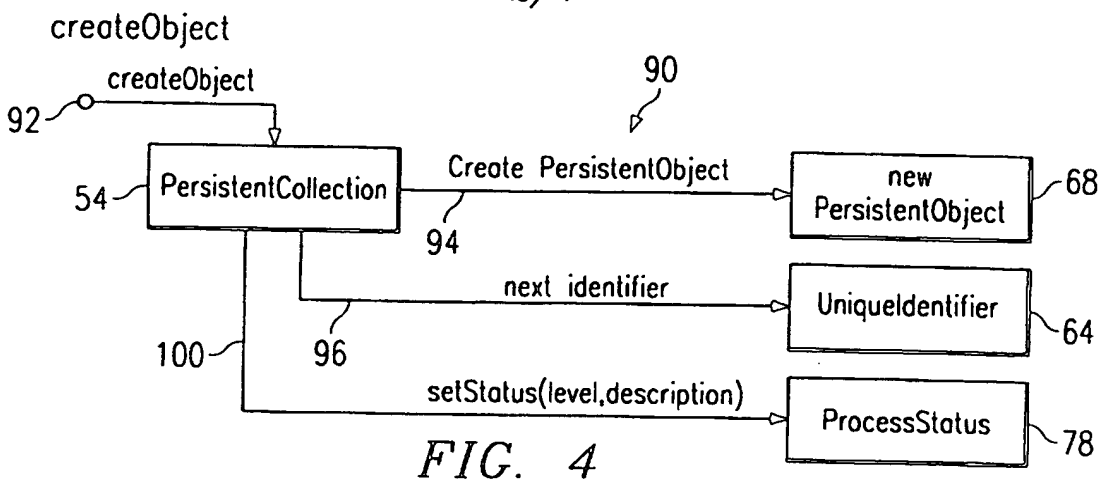


FIG. 4

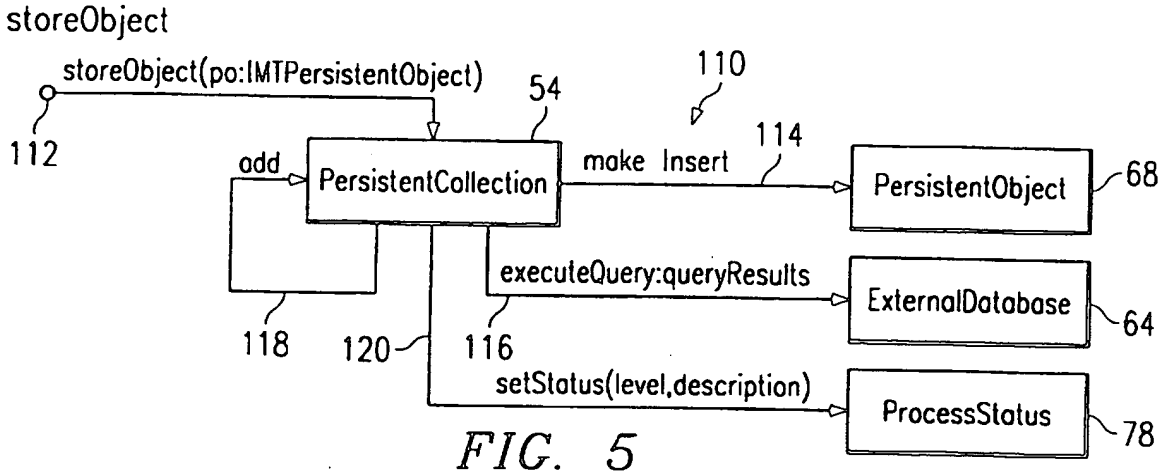


FIG. 5

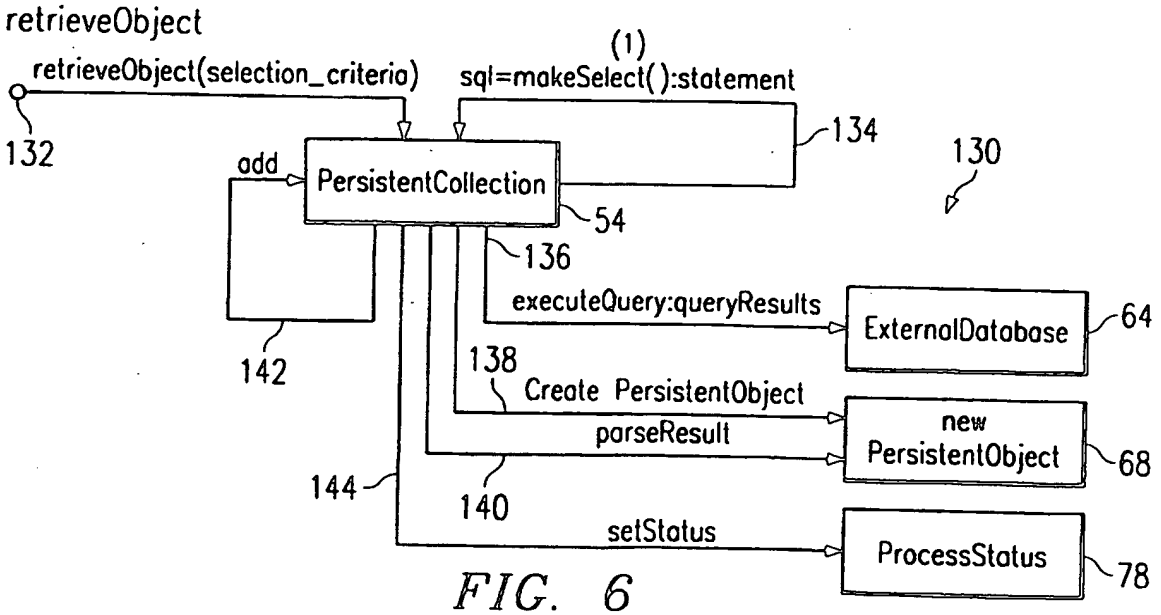


FIG. 6

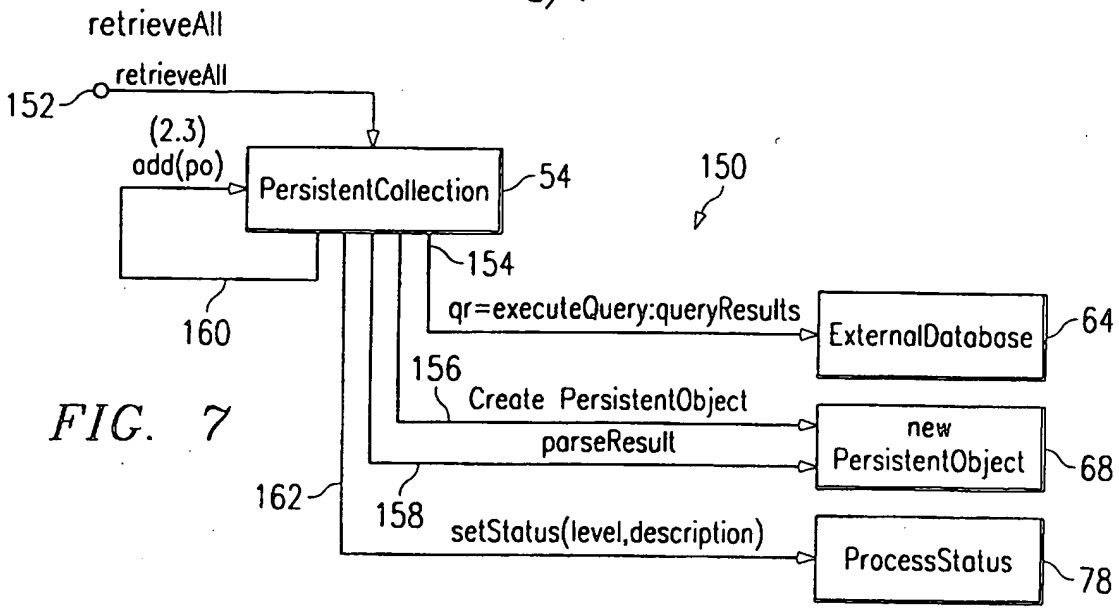


FIG. 7

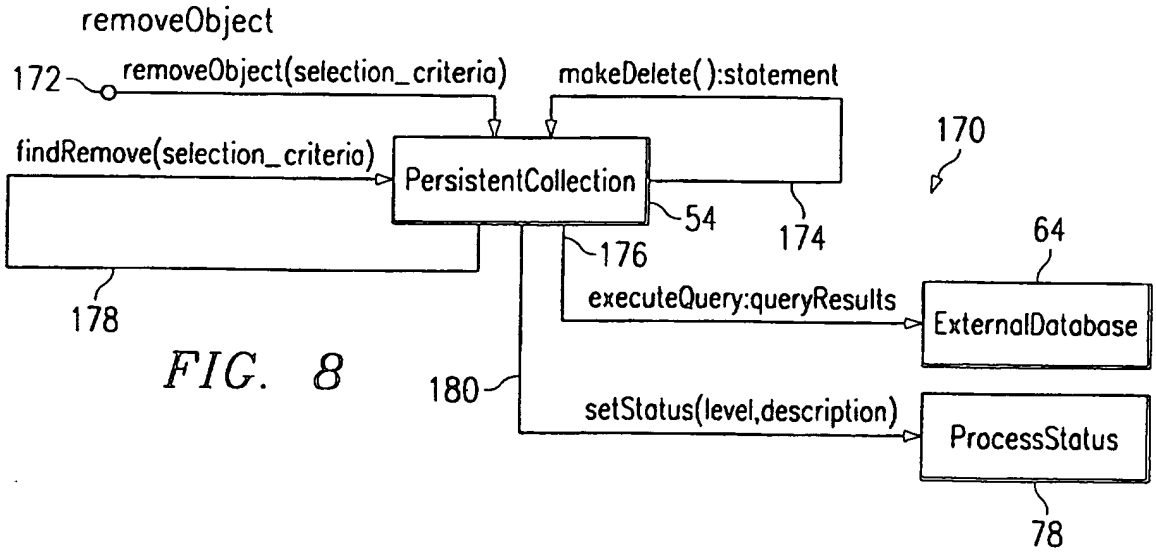


FIG. 8

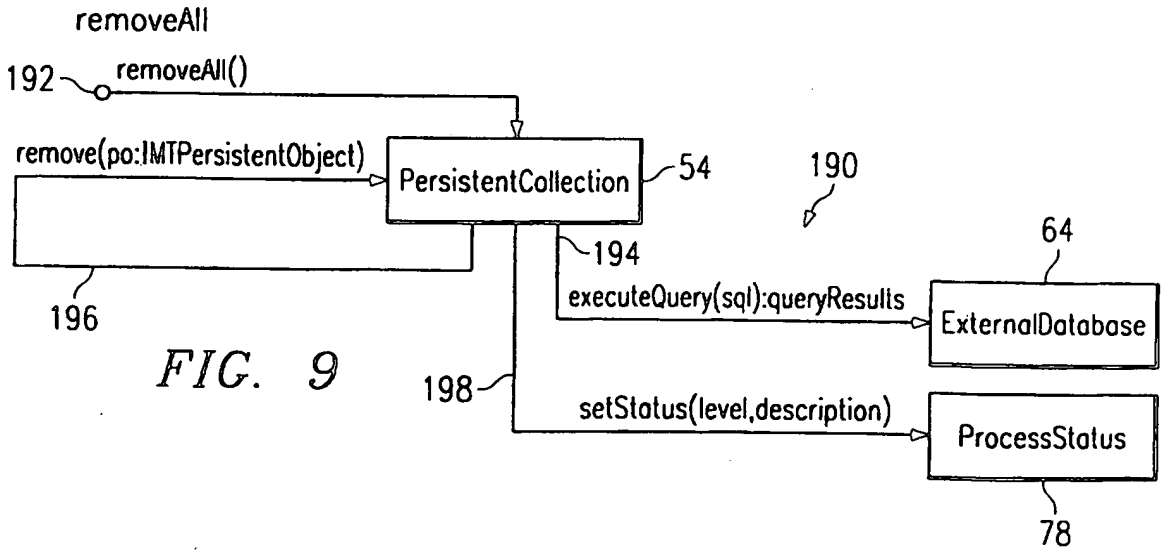


FIG. 9

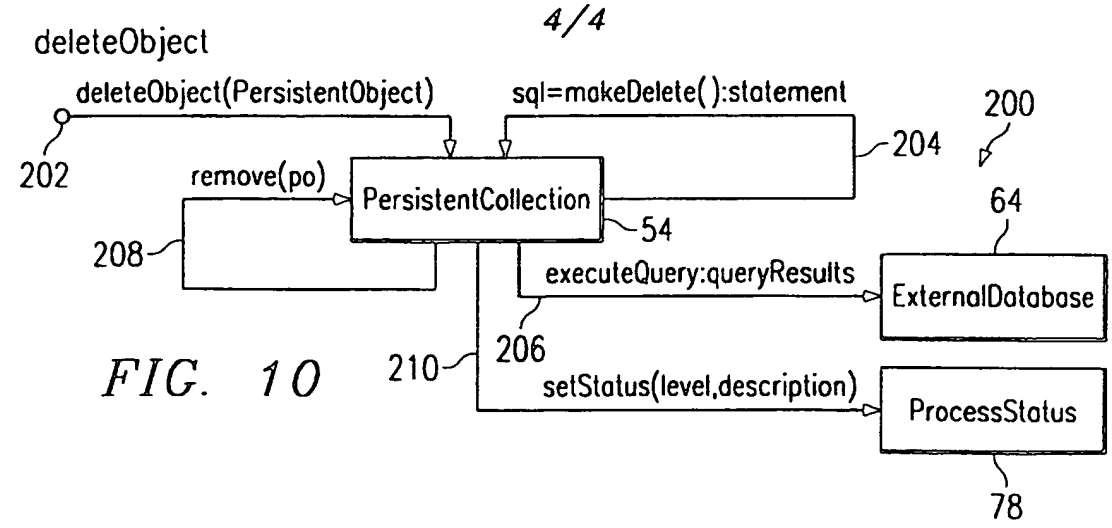


FIG. 10

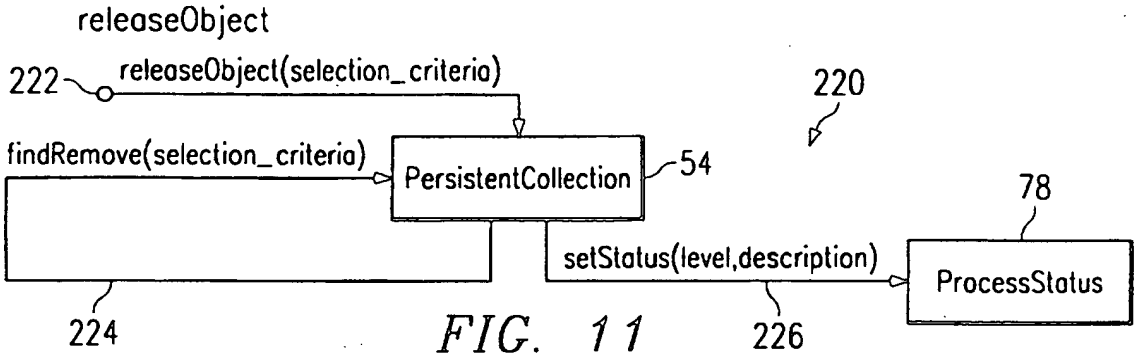


FIG. 11

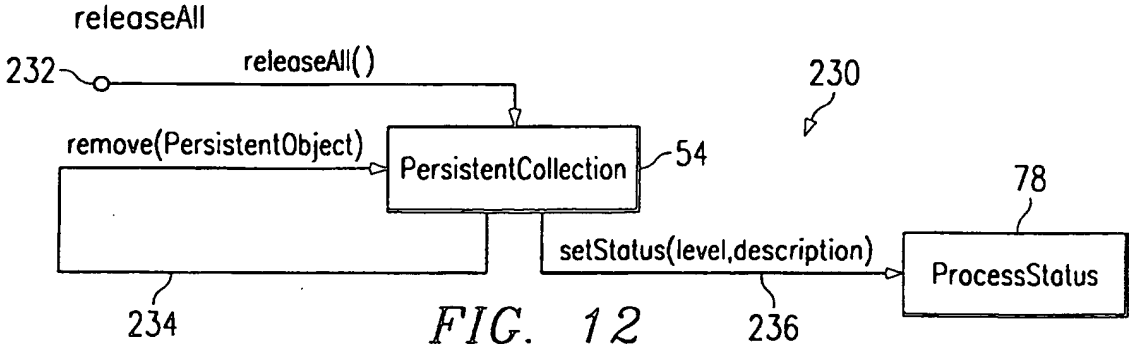


FIG. 12

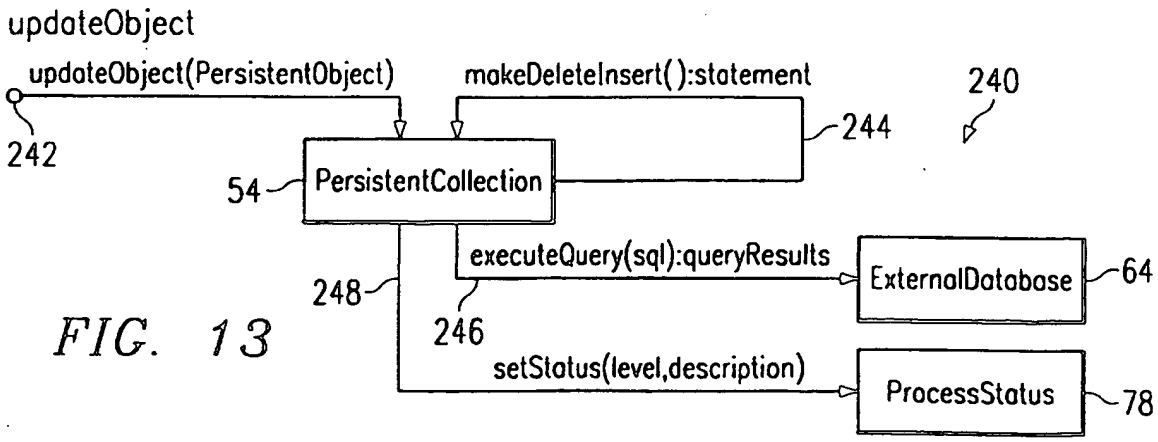


FIG. 13

INTERNATIONAL SEARCH REPORT

In tional Application No
PCT/US 97/07623

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F9/44

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 425 420 A (IBM) 2 May 1991 see column 3, line 19 - line 37 see column 4, line 2 - line 45 see column 8, line 1 - column 10, line 34	1-6, 8-14, 16-20
A	EP 0 631 229 A (IBM) 28 December 1994 see page 3, line 48 - line 54 see page 4, line 6 - line 46 see page 6, line 19 - line 45	1-20

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

1

Date of the actual completion of the international search	Date of mailing of the international search report
4 August 1997	29.08.97

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+ 31-70) 340-2040, Tx. 31 651 epo nl, Fax (+ 31-70) 340-3016	Brandt, J

Form PCT/ISA/210 (second sheet) (July 1992)

BNSDOCID: <WO___9742573A1_I_>

INTERNATIONAL SEARCH REPORT

Information on patent family members

In tional Application No
PCT/US 97/07623

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0425420 A	02-05-91	JP 3137730 A	12-06-91
		JP 8033862 B	29-03-96
		US 5247669 A	21-09-93
EP 0631229 A	28-12-94	JP 7098649 A	11-04-95

BNSDOCID: <WO___9742573A1_I_>